|  |
| --- |
| DEVOPS |
| Installation Notes – DNS/Web Servers/HAPROXY |

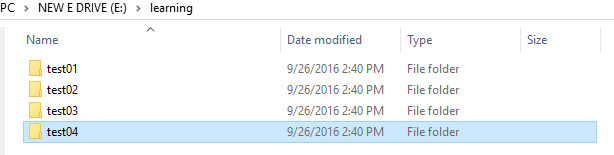
|  |
| --- |
|  |

# Setup of the instances

The details of the instances are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Machine | IP | OS | SSH Port | Purpose |
| test01.demo.org | 192.168.33.86 | Ubuntu 14.04 | 8222 | DNS Server, Puppet Master |
| test02.demo.org | 192.168.33.87 | Ubuntu 14.04 | 9222 | Jenkins Slaves  HA PROXY |
| test03.demo.org | 192.168.33.88 | Ubuntu 14.04 | 10222 | Production Tomcat Server |
| test04.demo.org | 192.168.33.89 | Ubuntu 14.04 | 11222 | Nagios Server |
| test06.demo.org | 192.168.33.91 | Ubuntu 14.04 | 13222 | Apache/NGINX |
| test07.demo.org | 192.168.33.92 | Centos 6.8 | 14222 | Apache/NGINX |

Initially you will create a folder structure similar to the following:



The names are arbitrary, but we do recommend that instance names have a pattern which can be scripted. A sample script to create this structure is given below (**Adapt as required**)

@echo off

mkdir e:\learning

for /l %%c in (1,1,4) do mkdir "e:\learning\test0%%c"

Now for each directory, you need to create a vagrant file inside. This creates a sample vagrant file inside each of these folders. A sample code is given below: (**Adapt as required**)

@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

cd "e:\learning\test0%%c"

vagrant init ubuntu/trusty64

)

cd e:\

Now for each folder, you need to edit the vagrant file to reflect correct

1. IP Address
2. SSH Port
3. Hostname

A sample vagrant file is attached for your reference.



After making the changes, you can start all machines by either manually going to each folder and typing the following at command prompt:

|  |
| --- |
| Vagrant up  @rem to stop a machine: use vagrant halt  **@rem the first time vagrant up is called, it has to download the full image so it will take time.** |

A sample script to start all of these machines is given below: (**Adapt as required**)

@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

cd "e:\learning\test0%%c"

vagrant up

)

cd e:\

A sample script to stop all of these machines is given below: (**Adapt as required**)

@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

cd "e:\learning\test0%%c"

vagrant halt

)

cd e:\

There is still one final step. You need to be able to SSH to these machines. To do so, create a folder called

E:\learning\keys. Then from each folder, navigate to folder: **.vagrant\machines\default\virtualbox**

Eg. *E:\learning\test01\.vagrant\machines\default\virtualbox*

Copy the file private\_key to e:\learning\keys giving it a unique name.

Eg. *copy e:\learning\test01\.vagrant\machines\default\virtualbox\private\_key e:\learning\keys*

Once this has been done, you can use any SSH client to connect to the instance as follows:

*Ssh -i e:\learning\keys\pk\_test01 -p 8222* [*vagrant@127.0.0.1*](mailto:vagrant@127.0.0.1)

A sample script to copy keys for all of these machines is given below: (**Adapt as required**)

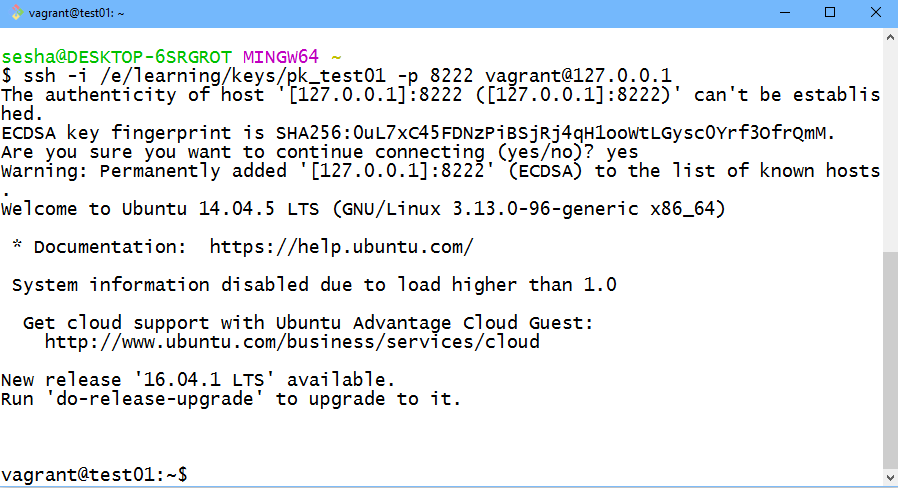
@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

copy e:\learning\test0%%c*\.vagrant\machines\default\virtualbox\private\_key e:\learning\keys\pk\_test0%%c*

)



# Install BIND/DNS on Ubuntu 14 (BOX#1)

To install Bind9 on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #1 as explained in previous section)

|  |
| --- |
| * Perform an update of repository   *sudo apt-get update*   * Install BIND software   *sudo apt-get install bind9 bind9utils bind9-doc dnsutils -y*   * Edit /etc/default/bind9 and add/modify line that contains OPTIONS to include support for IPV4   E,g. OPTIONS="-4 -u bind"   * Edit /etc/bind/named.conf.options and set any global level configurations. A sample named.conf.options is attached herwith      * Restart bind, Bind is UNIX implementation of DNS.   *sudo /etc/init.d/bind9 start*   * Verify core configuration using dig   dig @127.0.0.1 ubuntu.com   * Edit /etc/bind/named.conf.local. In our case, we are going to add zones for demo.org. So add the following lines to this file.   *zone "demo.org" {*  *type master;*  *file "/etc/bind/zones/db.demo.org";*  *#allow-transfer { 192.168.33.84; }; # ns2 secondary Name Server*  *};*  *zone "168.192.in-addr.arpa" {*  *type master;*  *file "/etc/bind/zones/db.192.168"; # 192.168.0.0/16 subnet*  *#allow-transfer { 192.168.33.84; }; # ns2 private IP address - secondary*  *};*   * Make the directory /etc/bind/zones and create empty files db.demo.org and db.192.168 under them. Sample content is attached below:   *sudo mkdir /etc/bind/zones*  *sudo touch /etc/bind/zones/db.demo.org*  *sudo touch /etc/bind/zones/db.192.168*     * Check Configuration.   *sudo named-checkconf*  *sudo named-checkzone sridemo.com db.sridemo.com*  *sudo named-checkzone 168.192.in-addr.arpa /etc/bind/zones/db.192.168*   * if all fine, restart   *sudo service bind9 restart*   * verify command   *dig demo.org @127.0.0.1*  *dig -tAXFR demo.org*  **Optionally: verify /etc/resolv.conf has your domain name and name server listed in it. This needs to be verified on each client.**  **If all is fine, a dig command should show output similar as below:** |

Edit /etc/resolvconf/resolv.conf.d/head and run *resolvconf -u*

# Install BIND/DNS on Centos

This is the same as above except you will use yum instead of apt-get. E.g

*sudo yum install bind bind-utils*

For enabling this to start on system boot,

*sudo systemctl enable named*

***The only other change is that instead of editing multiple files, you will edit the file: /etc/named.conf directly. All options are set in this file – it is still recommended to add zone files separately and include them inside /etc/named.conf e.g.***

***Include “/etc/named.edurekademo.com”***

For client configuration,

You can simply edit /etc/resolv.conf directly.

# Install Apache/NGINX

|  |
| --- |
| To install Apache on Ubuntu 14, just type in  ***sudo apt-get install apache2***  To install NGINX on Ubuntu 14, just type in  ***sudo apt-get install nginx***  To run both apache and NGINX at same time, either one of them should be edited to run on a different port other than the default 80. After making the change, just restart the services. E.g.  ***service [apache|nginx] restart***  To install NGINX or Apache on Centos 6.8, type in:  ***sudo yum install [httpd|nginx]***  If they do not start automatically, type  chkconfig [httpd|nginx] on |

# Install HA PROXY on Ubuntu 14.04

To install HA Proxy on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #2)

|  |
| --- |
| # Add relevant repositories  sudo add-apt-repository ppa:vbernat/haproxy-1.5  # Upgrade Repositories  sudo apt-get update  sudo apt-get dist-upgrade  #install HA Proxy  sudo apt-get install haproxy  # Configure HA Proxy...  #Edit /etc/haproxy/haproxy.cfg  #set maxconn to appropriate size.... This is done in global section  #in default section, under line mode http  # add follwing lines  option forwardfor  option http-server-close  #This will add X-Forwarded-For headers to each request, as well as reduce the latency between HAProxy and to preserve client persistent connections  # Add a frontend...  frontend http-frontend  bind 192.168.33.87:80 # This is the Public IP to which servers will connect  reqadd X-Forwarded-Proto:\ http  default\_backend wwwbackend  # Add backend  backend wwwbackend  server 1-www 192.168.33.91:80 check # First is to Ubuntu: 192.168.33.91  server 2-www 192.168.33.92:80 check # second is to Centos: 192.168.33.92  #server 3-www private\_ip\_3:80 check  #restart HA Proxy  sudo service haproxy restart |

<<END OF DOCUMENT>>